Senoguchi *J Big Data* (2021) 8:116
https://doi.org/10.1186/s40537-021-00506-x

Journal of Big Data

RESEARCH Open Access

Forecast of complex financial big data using model tree optimized by bilevel evolution strategy

Junsuke Senoguchi*

*Correspondence:
senoguchijs@stf.teu.ac.jp
Department of Computer Science, Tokyo University of Technology, Hachioji, Tokyo, Japan

Abstract

If a decision tree is constructed through a series of locally optimal solutions, such as the Greedy method, overfitting to the data is likely to occur. In order to avoid overfitting, many previous research have attempted to collectively optimize the structure of a decision tree by using evolutionary computation. However, if attributes of each split and their thresholds are searched simultaneously, the evaluation function becomes intermittent; thus, optimization methods assuming continuous distribution cannot be used. In this study, in order to enable efficient search assuming continuous distribution even for complicated data that contains a lot of noise and extraordinary values, such as financial time series data, the inner level search that optimizes each threshold value collectively given a specific attribute for each split in a model tree and the outer level search that optimizes the attributes of each split were performed by separate evolutionary computing. As a result, we obtained high prediction accuracy that far exceeded the performance of the conventional method.

Keywords: Evolutionary computing, Noisy function optimization, Financial market, Forecasting, Model tree

Introduction

A decision tree that recursively learns splits from higher to lower nodes is generally constructed by a series of local judgments and often involves the problem of noise overfitting. To solve this, various proposals have been made to optimize all splits collectively using evolutionary computation. However, many previous studies searched for the best tree by applying the branches of a highly rated tree to another tree. In the case of complicated data, i.e., data that contains a lot of noise and extraordinary values, the split with a high fitness value at one node is peculiar to the data sample at that node; the fitness value does not improve even if this is applied to another data sample [1, 2].

A method of searching all splits from one tree simultaneously without exchanging branches has also been proposed. However, because the number of combinations of features and their splitting points, or threshold values, is enormous, such a method is impractical for complicated data [3, 4].

Bilevel GA, which performs feature selection and searches for the order and thresholds of selected features by separate evolutionary computation, has been proposed recently as an efficient method for collectively optimizing the overall structure of a tree. In particular, this method has shown good estimation accuracy using relatively small-scale data [5–7].

However, the evaluation function of the tree becomes intermittent when a feature used for the split of the upper node changes owing to the simultaneous search of the feature order and the threshold value. Therefore, it is not possible to use a search method that presupposes continuity. To address this, a GA that repeats individual selection, crossover, and mutation is used as the search method because it optimizes splits that become non-deterministic polynomial time (NP)-hard when complex data are used. In addition, when employing such a GA, a split that is effective for an area region divided by specific upper nodes can be applied to a region divided by other upper nodes. Therefore, the fitness value of the tree does not always improve by generations, and the best tree might not be searched [8, 9]

In this study, we propose a method for searching splits on the premise of continuous distribution by searching for the optimal threshold under a specific feature and determining the optimal combination of features by separate evolutionary computation. In addition, we compare the computational time and prediction accuracy of this method with those using general machine learning and the results of previous studies using financial time series data as an example of complex real data.

Related work

Building a tree while avoiding overfitting

As mentioned in Sect. 2, when constructing a decision tree, there is a risk of falling into a locally optimal solution when using the greedy method, which recursively learns the splits from the upper to lower nodes. To avoid this, numerous attempts have been made to use evolutionary computation to find the optimal decision tree [10–15].

Attempts to collectively optimize the split of decision trees by genetic programming have been reported in the literature [16–19]. However, genetic programming is limited when streamlining optimization calculations by searching under continuous distribution. Moreover, it becomes difficult to converge to a global optimal solution when searching a decision tree with a large number of splits.

Building a tree by evolutionary computation

When optimizing the splits of the entire tree collectively by evolutionary computation, the fitness value of the tree changes non-linearly when the features of the splits of the upper node are changed. It is not possible to use evolutionary computation to generate individuals based on continuous distribution.

Optimization methods assuming continuous distribution include the steepest gradient method, Adam optimization, Newton's method, and Bayesian optimization [20, 21]. Such methods include one-point search, which has a higher probability to result in a locally optimal solution when handling complex data with a large number of dimensions. Under such conditions, approaches considered to be more suitable include the stochastic search method [22], which is a black box optimization method

using multipoint search, and real-valued evolutionary computation. Typical methods of real-valued evolutionary computation include real-valued GA [23], evolution strategy [24], differential evolution [25], and particle swarm optimization [26].

Real-valued GA showed high performance in the evaluation function with problems such as bad scale, intervariable dependency, and multimodality through minimal generation gap [27], unimodal distribution crossover (UNDX) [28], and real-coded ensemble crossover star (REXstar) [29]. However, none of these methods can handle multidimensional complex data that with large noise owing to their difficulty in adjusting the step size, population size, number of offspring, and other factors.

The covariance matrix adaptation evolution strategy (CMA-ES) [30] and distance-weighted exponential natural evolution strategies (DX-NES) [31] provide examples of the evolution strategy in which the need to adjust the step size, population size, and number of offspring is relatively small, which enables even a non-linear discontinuous evaluation function to be searched. However, the search performance of DX-NES deteriorates significantly when this strategy is applied to complex data with large noise [32].

CMA-ES, which is relatively resistant to noise, is considered to be desirable for the evaluation function that changes significantly by changing the threshold of the split in a decision tree. Of the many variations of CMA-ES proposed, [33] reported the best performance for data in noisy and uncertain environments, and the [34] model is suitable for searching decision trees.

Data and methods

Data

To compare the performance of a globally optimized model tree with that using general machine learning or a bilevel GA in a previous research, we evaluate the prediction accuracy using multiple benchmark data. Because the model tree constructed in this study aims to unravel a complicated data structure and extract a universal pattern for population, the data used for accuracy evaluation should also be applicable to complicated and noisy conditions.

UC Irvine Machine Learning Depository

From the UC Irvine Machine Learning Depository, we select relatively simple classification problems that have been used in many previous studies in addition to relatively complex regression problems meeting the following conditions.

- Explanatory variable type: continuous variable
- Data type: time series
- Number of explanatory variables: more than 10
- Number of samples: about 10,000 or more
- Low sparseness

Table 1 shows a summary of the data used in this study.

Table 1 Benchmark data used in this study

	# of features	# of cases
Classification		
Ecoli	8	336
Glass identification	10	214
Liver disorders	7	345
LSVT voice rehabilitation	126	309
Parkinson's disease	23	197
Connectionist bench	60	208
Wine	13	178
Regression		
Air quality	15	9,358
Appliances energy prediction	29	19,735
Electrical grid stability	14	10,000
Gas sensor array temperature	20	40,000
Real time election results	29	21,642

Financial market time series data

In addition, time series data of financial markets are used as actual data of complex systems. Financial market data contain many one-off factors and noise and serve as representative data for which high prediction accuracy cannot be obtained even by machine learning. This study uses as an objective variable the intraday return for the TOPIX Futures nearby month from the opening price at 08:45 to the closing price at 15:15. There are many other stock indices which represent the global financial markets, such as S&P 500, Dow Jones Industrial Average, Euro Stoxx 50 and so on; however, most of them have been on a consistent upward trend since 2009. Although TOPIX Futures have a smaller trading volume than other indices, they have no long-term trends and are a better example of complex system data. As explanatory variables, we select indicators that represent the financial markets of the United States and Japan. Unlike the objective variable, the explanatory variables do not necessarily have to be the price of the product that can actually be traded, although they need to reflect the movement of the entire financial market from a different perspective. For this reason, the change in the closing value of the stock index, exchange rate, and interest rate shown in Table 2 are used.

The financial data used to predict the TOPIX Futures in next business day include 4,901 intraday returns between January 04, 2001, and December 30, 2020 as an objective variable.

Methods

The purpose of this study is to obtain high prediction accuracy by a globally optimized model tree. For this purpose, a model tree is constructed by splitting a sample space with certain splits of a tree, evaluating the versatility of the pattern recognition model at each final node of the tree, and searching the best splits so the average versatility of the pattern recognition models in all final nodes becomes highest. The splits

Table 2 Financial data used to predict TOPIX Futures in next business day

Variables	Name	Change
Objective	TOPX Futures Nearby Month	Next Business Day Open ~ Close
Explanatory1	TOPIX	Previous Business Day Close ~ Current Business Day Close
Explanatory2	Nikkei 225/TOPIX	
Explanatory3	S&P 500	
Explanatory4	Dow Jones/S&P 500	
Explanatory5	JPY/USD	
Explanatory6	US Treasury 10Y Yield	
Explanatory7	US Treasury 5-30Y Yield Gap	
Explanatory8	TOPIX	3 Business Days Ago Close ~ Current Business Day Close
Explanatory9	Nikkei 225/TOPIX	
Explanatory10	S&P 500	
Explanatory11	Dow Jones/S&P 500	
Explanatory12	JPY/USD	
Explanatory13	US Treasury 10Y Yield	
Explanatory14	US Treasury 5-30Y Yield Gap	
Explanatory15	TOPIX	7 Business Days Ago Close ~ Current Business Day Close
Explanatory16	Nikkei 225/TOPIX	
Explanatory17	S&P 500	
Explanatory18	Dow Jones/S&P 500	
Explanatory19	JPY/USD	
Explanatory20	US Treasury 10Y Yield	
Explanatory21	US Treasury 5-30Y Yield Gap	

are not recursively searched individually; instead, they are simultaneously searched from large-scale combination optimization.

Bilevel GA

When optimizing the overall structure of a decision tree collectively, the fitness value of the tree changes discontinuously when the features used for the splits of a certain node change; therefore, the search method based on the continuity of functions cannot be used. In this study, we first identify the features to be used for each split and their positions in a tree randomly, and we then optimize the threshold for each feature by inner level search. Finally, we search the best features and their positions in a tree by outer level search. This method is referred to as a bilevel genetic algorithm (GA) used in this study. Figure 1 shows the outline of the bilevel GA by related works, and Fig. 2 shows the outline of the bilevel GA used in this study.

In the inner level search, the features used for each split and their positions in a tree are given, which enables the threshold value of each feature to be searched by a method using continuous distribution. All of the explanatory variables in this study are continuous; therefore, the threshold value change continuously. Because the data sample under a certain split changes discontinuously when the threshold values change, the evaluation function of the tree becomes discontinuous. However, data samples under certain splits change gradually one by one as the threshold changes, which enables the use of an evolutionary computation method that generates individuals based on continuous distribution.

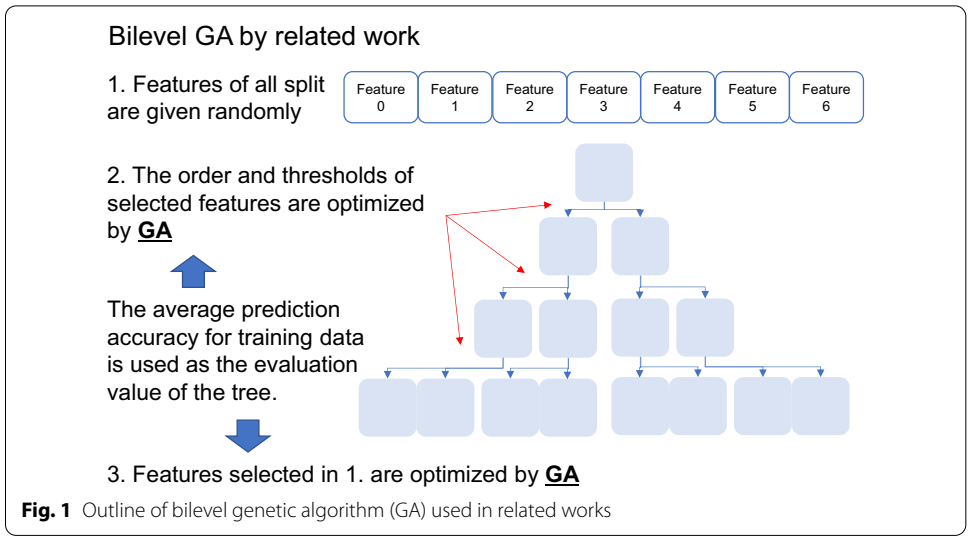

Fig. 1 Outline of bilevel genetic algorithm (GA) used in related works

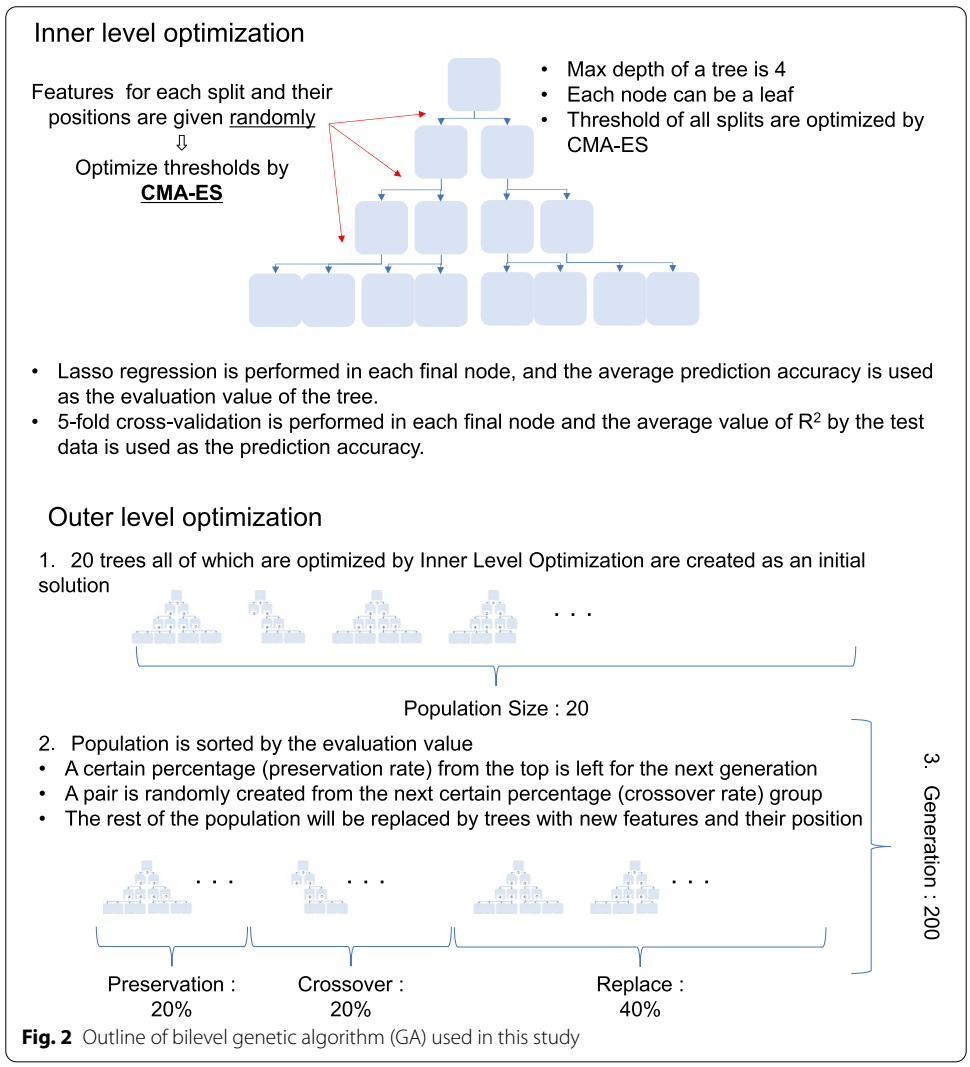

Fig. 2 Outline of bilevel genetic algorithm (GA) used in this study

In previous bilevel GA research, the position of the feature was also changed at the inner level, and the search method that presupposes the continuity of the function could not be used. Conversely, the inner level search in this study can be performed efficiently using continuous distribution.

In the outer level search, multiple trees with different features and their positions are generated randomly, and the optimal tree is searched by repeating the selection, crossover, and mutation. When evaluating each individual tree at the outer level, that in which all thresholds are already optimized according to the inner level is used.

Inner level optimization

CMA-ES, an evolutionary computation that performs multipoint search based on normal distribution, is used in this study to search the threshold value at the inner level search. This method updates the covariance matrix based on the evolutionary path that accumulates the previous solutions and generates offspring in the direction of movement of the solutions. It is suitable for the inner level search because it can search non-continuous evaluation functions by assuming normal distribution. In addition, CMA-ES is more resistant to noise than other efficient search methods that can handle discontinuous evaluation functions.

However, for data with particularly high levels of complexity such as financial time series data, the evaluation function becomes steep and multimodal, and the search for a global optimal solution requires a large amount of calculation even when using CMA-ES.

In general CMA-ES applications, the degree of freedom is $n + \frac{n^2-n}{2}$; the time complexity is $O(n^2)$; and the spatial complexity is $O(n^3)$ for the number of the dimension $O(n^2)$ of the evaluation function. By limiting the variance–covariance matrix $C^{(t+1)}$ used for individual genetion to diagonal components, the degree of freedom becomes n, and the amount of time and spatial complexity is reduced to $O(n)$:

$$c_{jj}^{(t+1)} = (1 - c_{cov})c_{jj}^{(t)} + \frac{1}{\mu_{cov}} c_{cov} \left(p_c^{(t+1)}\right)_j^2 + c_{cov}\left(1 - \frac{1}{\mu_{cov}}\right) \sum_{i=1}^{\mu} w_i c_{jj}^{(t)} \left(z_{i:\lambda}^{(t+1)}\right)_j^2, \quad j = 1, \ldots, n \tag{1}$$

where $C_{\mathrm{cov}} \in [0, 1]$ is the learning rate of diagonal element updates; $\frac{1}{\mu_{\mathrm{cov}}} \in [0, 1]$ is the weighting coefficient of the evolution path $p_c^{(t+1)}$; $z_{i:\lambda}^{(t+1)}$ is the i-th most rated of the $z^{(t+1)}$; and $\left(z_{i:\lambda}^{(t+1)}\right)_j$ is the i-th component of $\left(z_{i:\lambda}^{(t+1)}\right)_j$.

Outer level optimization

In the outer level search, the features used for each split and their positions are optimized. The parameters to be optimized are discrete values. In a decision tree, if the features used in a certain split are changed, the structure below will change significantly. Therefore, in the outer level search, the search method assuming continuous distribution cannot be used. For this reason, we use a GA that searches for individuals with high fitness values by repeating the selection, selection, crossover, and mutation because the shape of the evaluation function is not the issue.

As many trees in which thresholds of all splits are already optimized according to the inner level search as the population size are randomly generated as the initial population.

A certain percentage among them (preservation rate) is left for the next generation in the order of the fitness value of the tree, and a pair is randomly created from the next certain percentage (crossover rate) group. Branches at random positions are swapped in a pair, and the rest of the population will not be passed on to the next generation and will instead be replaced by trees with new features and their positions. This process is regarded as one generation, and the generation change is repeated. In this study, we use 20 population sizes, a 20% preservation rate, a 20% crossover rate, and 200 generations, as shown by the outer level search in Fig. 2.

Model evaluation method

In the classification problem, we use the weighted accuracy rate of each final node as the fitness value of the individual (model tree) generated by the bilevel GA. In the regression problem, we use the weighted prediction accuracy by linear regression analysis at the final node. The prediction accuracy is the average R^2 obtained by the five-fold cross-validation method. The linear regression model uses lasso regression with a regularization parameter of 0.1.

Large-scale combination optimization is required to select the optimal model tree when using such an evaluation method. Therefore, the Oakbridge-CX supercomputer system at the Information Technology Center, University of Tokyo, is used for the calculation.

Accuracy comparison of each method

To evaluate the prediction accuracy of each method, the results obtained from following approaches are compared: linear discriminant analysis; logistic regression analysis; support vector machine; neural network; classification and regression tree (CART); random forest; XGBoost; a decision tree constructed by bilevel GA proposed in [5], hereinafter referred to as bilevel GA by related work; and a decision tree constructed by bilevel GA proposed in this study, hereinafter referred to as bilevel GA by this study. For comparing the prediction accuracy used for comparison, the average classification accuracy rate of the results of five verifications is used according to the five-fold cross-validation method.

For regression problems, the following methods are used: multiple regression analysis, lasso regression analysis, partial minimum error, neural network, XGBoost, bilevel GA by related work, and a model tree constructed by bilevel GA proposed in this study, hereinafter also referred to as bilevel GA by this study. For the prediction accuracy used for comparison, the average R^2 according to the five-fold cross-validation method is used.

For problems using financial time series data, the following methods are used: multiple regression analysis, lasso regression analysis, partial minimum error, neural network, XGBoost, bilevel GA by related work, and bilevel GA by this study. For the prediction accuracy used for comparison, the average R^2 according to the five-fold cross-validation method is used.

Also, hyperparameters for all of the above machine learning methods are set by the three-fold cross validation grid-search.

Table 3 Accuracy comparison of classification problems

	E. coli	Glass identification	Liver disorders	Voice rehabilitation	Parkinson's disease	Connection bench	Wine
Linear discriminant analysis	13.1%	59.3%	64.4%	68.8%	85.7%	78.8%	97.8%
Logistic regression	88.1%	59.3%	66.7%	87.5%	83.7%	88.5%	100.0%
Support vector machine	86.9%	72.2%	67.8%	75.0%	93.9%	82.7%	100.0%
Multi-layer perceptron	87.3%	66.1%	71.4%	83.4%	87.1%	87.7%	100.0%
Decision tree	78.6%	60.2%	63.6%	77.5%	90.2%	76.3%	93.3%
XG boost	83.3%	75.9%	73.6%	75.0%	95.9%	88.5%	95.6%
Bilevel GA by related work	90.7%	74.4%	80.7%	75.6%	96.3%	80.8%	97.8%
Bilevel GA by this study	91.9%	79.3%	81.8%	78.8%	97.6%	84.6%	98.2%

Table 4 Accuracy comparison of regression problems

	Air quality		Energy predict		Electrical stability		Gas sensor		Election result	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Linear regression	0.49	0.45	0.16	0.16	0.65	0.64	0.58	0.59	0.98	0.98
Lasso regression	0.47	0.46	0.15	0.15	0.08	0.08	0.58	0.58	0.98	0.98
Partial minimum error	0.48	0.45	0.15	0.14	0.65	0.64	0.56	0.56	0.98	0.98
Neural network	0.65	0.61	0.57	0.30	0.87	0.85	0.67	0.61	0.60	0.60
XG boost	0.78	0.58	0.74	0.38	0.97	0.93	0.84	0.84	1.00	1.00
Bilevel GA by related work	0.50	0.49	0.59	0.48	0.72	0.60	0.70	0.64	0.95	0.93
Bilevel GA by this study	0.79	0.69	0.76	0.61	0.98	0.95	0.91	0.89	1.00	1.00

Results and discussion

When using the UC Irvine data in a relatively simple classification problem, bilevel GA by this study showed high prediction accuracy, as did the other methods. In regression problems with a large numbers of data, the prediction accuracy of bilevel GA by this study exceeded that of other methods. Moreover, this method showed a much higher estimation accuracy than that of other methods when using financial data.

Prediction accuracy

For the classification problems of UC Irvine, bilevel GA by this study showed high prediction accuracy, as did the other methods (Table 3). Because the other methods also showed relatively high prediction accuracy, the pattern was easily recognized.

For the regression problems of UC Irvine, bilevel GA by this study showed better prediction accuracy than other methods (Table 4). Some of other methods showed relatively low prediction accuracy because the patterns were difficult to recognize in some data, although the bilevel GA by this study was high even for such complicated problems.

Moreover, in the financial time series data, this method showed a prediction accuracy that greatly exceeded that of other methods (Table 5). Although many studies have been conducted on predicting financial market prices by machine learning, no clear

Table 5 Accuracy comparison using financial time series data

	Train	Test
Linear regression	0.49	0.02
Lasso regression	0.49	0.07
Partial minimum error	0.45	0.06
Neural network	0.68	0.06
XGBoost	0.72	0.09
Bilevel GA by related work	0.48	0.01
Bilevel GA by this study	0.55	0.35

conclusion has been reached. However, bilevel GA by this study showed explicitly higher accuracy than that using general machine learning.

In addition, bilevel GA by this study uses elitism for the outer level search; thus, the fitness of trees improves by generations. Figure 3 shows the transition of the fitness of a best model tree constructed by bilevel GA by this study in Table 5 for each generation.

Furthermore, in the model tree optimized at the outer level, which is the best tree selected by using the bilevel GA by this study, we confirmed the fitness of the tree for each generation at the inner level search (Fig. 4). Because the best solution is not preserved at the inner level search, the fitness of the tree does not always increase with each generation. As indicated in Fig. 4, although the shape of the evaluation function exhibits many irregularities, a global optimum solution was obtained. Therefore, CMA-ES has succeeded in searching a global optimum solution even though the fitness fluctuated owing to the influences of noise and extraordinary values.

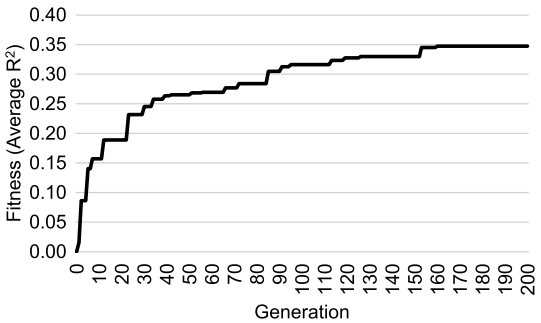

Fig. 3 Changes in tree fitness for each generation at the outer level

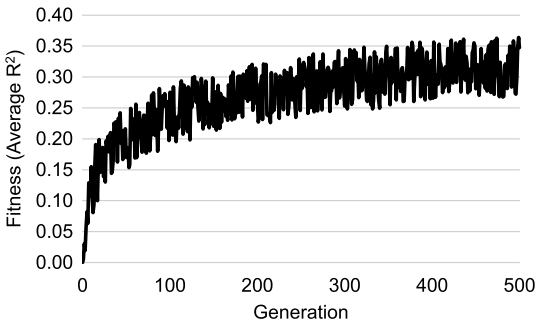

Fig. 4 Changes in tree fitness for each generation at the inner level

Impact of crossover

In this study, we used 20 population sizes, a 20% preservation rate, a 20% crossover rate, and 200 alternations of generations for outer level optimization, as discussed in Sect. 3.2.3.

By using the best tree in Table 5, the outer level search was performed using different crossing rates. The optimal fitness of the tree decreased as the crossover rate increased. Moreover, the fitness did not change significantly when the crossover rate was reduced (Table 6), and the change in the prediction accuracy was small even when values other than the crossover rate changed.

As discussed in Sect. 1, the split at the lower node is effective only for the subsample generated by dividing the total sample at the upper level, which explains why the relatively high crossover rate deteriorated the fitness.

Conclusions

Decision trees have been widely used for data analysis because of their ease of interpretation. However, if the Greedy method, which recursively searches for split from the upper node to the lower node, is used, overfitting is likely to occur because a tree is constructed with a series of locally optimal solutions. Many previous research has attempted to collectively optimize the structure of a decision tree using evolutionary computation; however, many of them searched attributes of each split and their thresholds simultaneously; thus, optimization methods assuming continuous distribution cannot be used. In this study, we proposed bilevel GA that improved the problems in the previous research. As a result, we found that it surpassed the conventional methods in terms of performance from relatively simple problems to complex problems.

In this study, I compared the proposed method with the several major methods from many existing machine learnings and evaluated its superiority. It cannot be denied that some of the existing machine learning methods may exceed the proposed method in the prediction accuracy. On the other hand, the proposed method was proven to be able to improve the prediction accuracy of complex data while taking advantage of the fact that the prediction model is a decision tree with which the interpretation is easy. This can be said to be the great significance of this study.

We also found that it was still not possible to derive sufficiently high prediction accuracy even by using the proposed bilevel GA, when data with increased complexity, such

Table 6 Impact on performance for each crossover rate

Crossover rate	R^2
0%	0.027
10%	0.165
20%	0.348
30%	0.198
40%	0.020
50%	0.017
60%	0.008
70%	0.006

as financial time series data, was used. The reason for this is thought to be that the tree was constructed based on the fitness of all final nodes of the tree. We may obtain a better prediction result if a tree is constructed so some final nodes with an extremely low fitness are excluded from the evaluation of the tree and if data classified as the final nodes excluded from the evaluation are not subject to prediction.

Also, in this study, we used a binary tree; however, the space in which pattern recognition model perform well is not necessarily all on one side from a certain threshold of the entire sample data. Therefore, it is desirable to use a multi-way tree when dividing the space by a tree. We want to make these issues for the future task.

Acknowledgements
Not applicable.

Authors' contributions
JS took on the main role performed the literature review, implemented the proposed model, conducted the experiments and wrote manuscript.

Funding
The authors declare that they have no funding.

Availability of data and materials
All data used in this study are publicly available and accessible in the below sources.
Benchmark data: http://archive.ics.uci.edu/ml/index.php.
Financial data: https://finance.yahoo.com/.

Declarations

Ethics approval and consent to participate
The authors Ethics approval and consent to participate.

Consent for publication
The authors consent for publication.

Competing interests
The authors declare that they have no competing interests.

Received: 13 June 2021 Accepted: 20 August 2021
Published online: 03 September 2021

References

1. Papagelis A, Kalles D. Breeding decision trees using evolutionary techniques. In: Eighteenth international conference on machine learning. Morgan Kaufmann Publishers, Inc.; 2001. p. 393–400.
2. Fu Z, Golden BL, Lele S, Raghavan S, Wasil EA. A genetic algorithm-based approach for building accurate decision trees. INFORMS J Comput. 2003;15(1):3–22. https://doi.org/10.1287/ijoc.15.1.3.15152.
3. Shirasaka M, Zhao Q, Hammami O, Kuroda K, Saito K. Automatic design of binary decision trees based on genetic programming. In: Second Asia-Pacific conference on simulated evolution and learning. 1998.
4. Zhao Q, Shirasaka M. A study on evolutionary design of binary decision trees. In: IEEE congress on evolutionary computation. 1999. p. 1988–1993.
5. Adibi MA. Single and multiple outputs decision tree classification using bi-level discrete-continues genetic algorithm. Pattern Recogn Lett. 2019;128:190–6. https://doi.org/10.1016/j.patrec.2019.09.001.
6. Dhebar Y, Deb K. Interpretable rule discovery through bilevel optimization of split-rules of nonlinear decision trees for classification problems. IEEE Trans Cybern. 2020. https://doi.org/10.1109/TCYB.2020.3033003.
7. Hanh PTH, Thanh PD, Binh HTT. Evolutionary algorithm and multifactorial evolutionary algorithm on clustered shortest-path tree problem. Inf Sci. 2021;553:280–304. https://doi.org/10.1016/j.ins.2020.10.024.
8. Tanigawa T, Zhao Q. A study on efficient generation of decision trees using genetic programming. In: Genetic and evolutionary computation conference. 2000. p. 1047–1052.
9. Aitkenhead MJ. A co-evolving decision tree classification method. Expert Syst Appl. 2008;34(1):18–25. https://doi.org/10.1016/j.eswa.2006.08.008.
10. Safavian SR, Landgrebe D. A survey of decision tree classifier methodology. IEEE Trans Syst Man Cybern. 1991;21(3):660–74. https://doi.org/10.1109/21.97458.
11. Murthy SK. Automatic construction of decision trees from data: a multi-disciplinary survey. Data Min Knowl Disc. 1998;2(4):345–89. https://doi.org/10.1023/A:1009744630224.
12. Freitas AA. A critical review of multi-objective optimization in data mining: a position paper. ACM SIGKDD Explorations Newsl. 2004;6(2):77–86. https://doi.org/10.1145/1046456.1046467.

13. Rokach L, Maimon O. Top-down induction of decision trees classifiers—a survey. IEEE Trans Syst Man Cybern Part C. 2005;35(4):476–87. https://doi.org/10.1109/TSMCC.2004.843247.
14. Espejo PG, Ventura S, Herrera F. A survey on the application of genetic programming to classification. IEEE Trans Syst Man Cybern Part C. 2010;40(2):121–44. https://doi.org/10.1109/TSMCC.2009.2033566.
15. Barros RC, Basgalupp MP, de Carvalho ACPLF, Freitas AA. A survey of evolutionary algorithms for decision tree induction. IEEE Trans Syst Man Cybern Part C. 2012;42(3):291–312. https://doi.org/10.1109/TSMCC.2011.2157494.
16. Burgess CJ, Lefley M. Can genetic programming improve software effort estimation? A comparative evaluation. Inf Softw Technol. 2001;43(14):863–73. https://doi.org/10.1016/S0950-5849(01)00192-6.
17. DeLisle RK, Dixon SL. Induction of decision trees via evolutionary programming. J Chem Inf Comput Sci. 2004;44(3):862–70. https://doi.org/10.1021/ci034188s.
18. Zhao H. A multi-objective genetic programming approach to developing pareto optimal decision trees. Decis Support Syst. 2007;43(3):809–26. https://doi.org/10.1016/j.dss.2006.12.011.
19. To C, Pham T. Analysis of cardiac imaging data using decision tree based parallel genetic programming. In: 6th international symposium on image and signal processing and analysis. 2009. p. 317–320.
20. Shahriari B, Swersky K, Wang Z, Adams RP, de Freitas N. Taking the human out of the loop: a review of bayesian optimization. Proc IEEE. 2016;104(1):148–75. https://doi.org/10.1109/JPROC.2015.2494218.
21. Adams RP, Stegle O. Gaussian process product models for nonparametric nonstationarity. In: International conference on machine learning. 2008. p. 1–8.
22. Larraga RE, Lozano JA, Pena JM. A review of cooperation between evolutionary computation and probabilistic graphical models. In: Second symposium on artificial intelligence CIMAF. 1999. p. 314–324.
23. Davis L. The handbook of genetic algorithms. New York: Van Nostrand Reinhold; 1990.
24. Beyer HG, Schwefel HP. Evolution strategies: a comprehensive introduction. Nat Comput. 2002;1(1):3–52. https://doi.org/10.1023/A:1015059928466.
25. Storn R, Price K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim. 1997;11(4):341–59. https://doi.org/10.1023/A:1008202821328.
26. Kennedy J, Eberhart RC. Particle swarm optimization. In: IEEE international joint conference on neural networks. 1995. p. 1942–1948
27. Sato H, Ono I, Kobayashi S. A new generation alternation model of genetic algorithms and its assessment. J Jpn Soc Artif Intell. 1997;12(5):734–44.
28. Ono I, Kobayashi S, Yoshida K. Optimal lens design by real-coded genetic algorithms using UNDX. Comput Methods Appl Mech Eng. 2000;186(2–4):483–97. https://doi.org/10.1016/S0045-7825(99)00398-9.
29. Kobayashi S. The frontiers of real-coded genetic algorithms. J Jpn Soc Artif Intell. 2009;24(1):128–43.
30. Hansen N, Ostermeier A. Completely derandomized self-adaptation in evolution strategies. Evol Comput. 2001;9(2):159–95. https://doi.org/10.1162/106365601750190398.
31. Fukushima N, Nagata Y, Kobayashi S, Ono I. Proposal of distance-weighted exponential natural evolution strategies. In: IEEE congress on evolutionary computing. 2012. p. 164–170.
32. Masutomi K, Nagata Y, Ono I. A novel evolution strategy for noisy function optimization. Trans Jpn Soc Evol Comput. 2015;6(1):1–12. https://doi.org/10.11394/tjpnsec.6.1.
33. Hansen N, Niederberger ASP, Guzzella L, Koumoutsakos P. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Trans Evol Comput. 2009;13(1):180–97. https://doi.org/10.1109/TEVC.2008.924423.
34. Richter SN, Schoen MG, Tauritz DR. Evolving mean-update selection methods for CMA-ES. In: Evolutionary computation conference. 2019. p. 1513–1517.

Publisher's Note